The current issue and full text archive of this journal is available on Emerald Insight at:
www.emeraldinsight.com/2399-6439.htm

Received 28 August 2017
Revised 15 November 2017
Accepted 18 November 2017

Lifting and shifting the Air Force retail supply system

Tommie L. Ellis, Robert A. Nicholson, Antoinette Y. Briggs, Scott A. Hunter and James E. Harbison
Air Force Life Cycle Management Center/HIAR, US Department of the Air Force, Maxwell AFB (Gunter Annex), Alabama, USA

Paul S. Saladna
Department of Information Technology, NTT Federal Data Systems, Stone Mountain, Georgia, USA, and

Michael W. Garris, Robert K. Ohnemus, John E. O'Connor and Steven B. Reynolds
Air Force Life Cycle Management Center/HIAR, US Department of the Air Force, Maxwell AFB (Gunter Annex), Alabama, USA

Abstract

Purpose – Rising operational costs and software sustainment concerns have driven the Air Force to move to newer technology to ensure that the Air Force Standard Base Supply System (SBSS) can continue to provide affordable and sustainable mission support in the years to come. This paper aims to summarize the successful software modernization effort the Air Force undertook to achieve that objective.

Design/methodology/approach – The paper describes the preliminary system updates that were required to isolate the SBSS software from all internal and external system and user interfaces in preparation for the subsequent successful code roll effort. Once the legacy SBSS component was fully isolated, the SBSS software modernization objective was achieved via a "code roll" conversion of the SBSS software from legacy COBOL to Java code, and movement of the integrated logistics system-supply application from a proprietary information technology (IT) platform to an open IT operating environment.

Findings – The SBSS system modernization yielded immediate and significant IT operational cost reductions and provided an important foundation for achieving Air Force logistics system consolidation and cloud computing objectives going forward.

Originality/value – The SBSS modernization experience should be useful in assisting similar data system software modernization efforts.

Keywords Java, COBOL, Integrated logistics system-supply (ILS-S), Open-source environment, Re-platform, Standard base supply system (SBSS)

Paper type General review

Introduction

Mainframe computers and second- and third-generation software have been the heart and soul of the integrated logistics system-supply (ILS-S) for over 50 years. The Air Force Standard Base Supply System (SBSS), a component of the ILS-S, is actively used by over

Journal of Defense Analytics and Logistics
Vol. 1 No. 2, 2017
pp. 172-184
Emerald Publishing Limited
2399-6439
DOI 10.1108/JDAL-08-2017-0016

© In accordance with section 105 of the US Copyright Act, this work has been produced by a US government employee and shall be considered a public domain work, as copyright protection is not available. Published in *Journal of Defense Analytics and Logistics*. Published by Emerald Publishing Limited.

100,000 Air Force personnel to enable the ordering, stocking, storing and daily management of worldwide operational supply needs. The SBSS component was among the US Air Force's earliest logistics data systems. The system was originally fielded as a program assembler language (PAL)-based application in 1965 and operated on mainframe computers that filled the basements of large office buildings at most Air Force bases. As software technology improved, the Air Force transitioned the SBSS system software to common business-oriented language (COBOL) to improve the system's viability and sustainability. Unisys mainframe hardware platforms have served the Air Force well in hosting the COBOL SBSS application over the years; however, the operational expense of processing the huge number of worldwide user system transactions has steadily increased. The time has come to move on to a more robust and affordable information technology (IT) foundation for the Air Force base-level supply system.

IT systems experts from the Air Force and industry agreed "[. . .] if agencies do not take the appropriate steps toward modernization, they will almost certainly incur increased costs related to performance and maintenance" (Dittmer, 2013, p. 2). SBSS operational sustainment worries were further exacerbated by a continued decrease in the availability of skilled COBOL software programmers. These two issues taken together have been the source of significant concern to Air Force IT leaders since the early 1990s. Thanks to a forward-thinking IT team, the system has now been modernized to provide a much more affordable and sustainable foundation for the next 50 years of supply support for Air Force operations. The purpose of this article is to briefly retrace the SBSS's implementation path from 1965 to today and describe the ILS-S program office's successful IT system modernization initiative.

Early days of the Standard Base Supply System

Some of us remember 1965. Technology was exploding across the spectrum. Kids played with battery-operated motorized Tinkertoy sets, and we listened to the Beatles on our pocket-sized transistor radios (The People History, 2017). The Gateway Arch was completed in St Louis, and that convertible Mustang we all wish we owned today sold new for $2,663 (CJ Pony Parts, 2017). Three successful Gemini missions laid the groundwork for an eventual manned mission to the moon. At the same time, US military operations in Vietnam were escalating. President Johnson ordered Operation Rolling Thunder with the intent to destroy the North Vietnamese supply network and air defense system, and Air Force logistics operations were going full tilt, delivering supplies in support of those operational missions throughout the region (Connolly, 2017).

In September 1965, following 2.5 years of development, the Air Force Supply System Design Office (SSDO) began fielding the SBSS at ten Air Force bases per month. By August 1966, 110 bases were converted to the new system, all with identical UNIVAC 1050-II computers, programs and procedures. The implementation of the SBSS was a significant leap forward in the evolution of Air Force supply support. Before SBSS, there was no standardization among base supply processes and procedures. A 1968 Rand Corporation study stated that until the SBSS was implemented:

> [. . .] bases operated under a variety of computer and punch card procedures; thus, the previous base experience with electronic data processing equipment varied widely among commands. The approaches to training and other preparation also differed from one command to another; and, of course, the background of base personnel varied within commands. (Codlin *et al.*, 1968, p. 2).

The PAL-based application was implemented at every base in the Air Force on SBSS dedicated UNIVAC 1050-II computer mainframes. While the PAL-based UNIVAC implementation of a standard system was a significant advancement at the time, there were

important disadvantages to software applications developed with assembly languages. Assembly languages are just one generation removed from the first-generation machine languages that represented all data and program instructions as binary 1 and 0 digits corresponding to "on" and "off" electrical states. Assembly language enabled programmers to use codes and key words to represent the binary digits and data storage locations, but a machine-specific translator was required to convert those key words into machine language. A key disadvantage of early programing languages was that they required a great amount of detail, and as a result, writing assembly language code was repetitive, tedious and error-prone (Wolfe, 2017).

In the late 1960s, the Department of Defense (DoD) mandated the use of a new standardized high-level programing language called COBOL, which was more conducive to supporting business needs. Given that mandate, the SSDO began the line-by-line conversion of the SBSS software from PAL to COBOL 74 in 1981, completing and fielding the COBOL version of the system in 1983. That implementation also involved the migration of the SBSS application from the dedicated "supply-only" UNIVAC 1050-II to a Unisys 1100 platform that was shared with other base activities (Spruce, 2002). An unfortunate consequence of the COBOL implementation was that the Air Force was paying a high cost for operational transaction processing and data storage on the proprietary Unisys 1100 platform. That "pay-as-you-go" agreement became increasingly expensive as the years progressed. That high cost arrangement continued after the Air Force migrated the COBOL 74 code to COBOL 85 in the early 1990s.

The way forward

The supply management capabilities provided by the ILS-S/SBSS will obviously continue to be fundamental to Air Force mission support. It is vitally important for the Air Force to exploit IT improvement opportunities to ensure that the system can continue to provide those critical supply management capabilities in the most sustainable and cost-effective way possible going forward. The key issues driving the need to modernize the ILS-S/SBSS component are high operational costs and system software sustainability concerns.

High operational costs

In general, there are two categories of enterprise technology stacks: proprietary and open source. Proprietary frameworks are closed source and thus vendor-dependent. Additionally, proprietary frameworks require the payment of private license fees to use the framework. With proprietary frameworks, the costs of setting up content management systems are often considerably higher than open-source options, and a contract is frequently required. Casson and Ryan (2006) suggest that systems that are locked into a single provider run the risk of a non-competitive pricing structure and technological or vendor obsolescence. Open-source frameworks on the other hand are not private; they are free and available for download and editing by anyone. With open-source software, users can make broad changes without having to consider proprietary contractual/cost issues and port the software to new operating systems and instruction set architectures. Additional important benefits of open-source frameworks are improved security, affordability, transparency, interoperability and flexibility. As stated earlier, the ILS-S legacy SBSS component is operated in a proprietary Unisys environment. The preliminary analysis conducted by the ILS-S program office concluded that the Air Force could realize a $20M to $25M annual reduction in operating costs by moving off the Unisys proprietary framework to a lower-cost, mid-tier open-source environment.

System software sustainability

COBOL is an older software language that is no longer widely used for the development of new applications in the commercial world. In fact, many universities have stopped offering COBOL courses in their computer programing curricula because young developers choose to prepare for the job market by developing expertise in more contemporary software languages (Logapps, LLC, 2015). Consequently, the pool of COBOL programmers that would be required for the sustainment of the legacy SBSS system is aging and continuing to decrease, which is driving up the development and maintenance labor costs of COBOL-based systems. In fact, 59 per cent of current COBOL developers are over the age of 45, and 93 per cent are over 35 years old. This shrinking pool of skilled COBOL programmers is a significant concern with respect to sustainability of the legacy of ILS-S/SBSS component in the coming years.

Alternatives to the proprietary Unisys platform

The Air Force has been keenly aware of the operational cost and software sustainment issues associated with the Unisys COBOL implementation of the SBSS since the early 1990s. Therefore, the Air Force pursued four (ultimately unsuccessful) efforts to replace the SBSS application between the early 1990s and 2014 to provide a more affordable and supportable option than the proprietary Unisys COBOL implementation (Ohnemus, 2004).

Recode to ADA

After the SBSS software was successfully upgraded from COBOL 74 to COBOL 85, the ILS-S program office decided to move the COBOL code to the ADA programing language so the system could be subsequently moved to a more open operating system environment. However, developmental complications arose, and the effort was abandoned.

Government Online Data

The next significant endeavor to move away from Unisys COBOL involved a 1997 initiative to implement a commercial-off-the-shelf (COTS) product called GOLD (Government Online Data), a software solution that could replace the functionality of the SBSS. The Air Force expectation was that the COTS product would provide a 70-80 per cent solution in replacing the SBSS software (Olsen, 1997). However, the GOLD software ultimately implemented only 27 per cent of the existing SBSS functionality (Vaas, 2001).

Functional capability component development

After GOLD was abandoned, the Air Force set out to incrementally replace the SBSS by developing functional software components. The end result was to be a single system (versus over 300 separate SBSS functional programs), with most of the existing business rules remaining intact. This approach initially showed promise. The user interfaces were Web-enabled using enterprise application integration (EAI) principles via small Java applications (applets) in December 2000, and a consolidated Oracle database was fully fielded via a Solaris environment UNIX platform in 2003. However, the plan was suspended when the United States Air Force (USAF) decided to pursue a commercial enterprise resource planning (ERP) solution.

Expeditionary combat support system

In 2004, the Air Force conceived the implementation of an ERP system called the expeditionary combat support system (ECSS). The Air Force expectation for the ECSS was

to provide end-to-end transformation by replacing more than 420 legacy Air Force logistics data systems serving over 250,000 users (Bliss, 2013). In 2006, the Air Force awarded a contract for the implementation of the Oracle ERP suite. By 2009, there were sufficient concerns about the requirements of blueprinting efforts to warrant a program restructure, and then another program restructure occurred in October 2010. By 2012, the program had spent $1.03bn and, by most assessments, had delivered very little capability (Stross, 2012). In fact, Condliffe (2012) contends that at the time the program was cancelled in November 2012, the ECSS program had delivered only one-fourth of the originally planned capabilities. A key shortcoming was the inability of the ECSS application to meet the fiscal year 2017 DoD Financial Improvement Audit Readiness (FIAR) requirements.

At the same time the Air Force was pursuing software modernization alternatives, a significant drawdown of Air Force manning was underway. As shown in Figure 1, Air Force manpower began a marked manning decrease beginning in about 1989 (Coleman, 2017). That manpower decrease affected the Air Force's Supply career field manning in a way that necessitated the development of less manpower-intensive, centralized supply computer operations. However, given the Air Staff's 2012 decision to cancel the ECSS program, it was not clear how those enterprise supply management capabilities were going to be enabled.

Enterprise solution-supply

During the ECSS implementation effort, the ILS-S program office was conducting an ongoing proof of concept (POC) effort to exploit emerging EAI principles within the ILS-S application. The ILS-S EAI initiative, called the enterprise solution-supply (ES-S), was initially undertaken in late 2003, even before the ECSS initiative was started in 2004 and, fortunately, was continued during the Air Force's pursuit of the ECSS implementation. ES-S matured during the POC as a Java-based, comprehensive, Web-enabled user interface to the legacy SBSS application and was successfully implemented as a component of the ILS-S application on a modern Unix-based platform outside the proprietary Unisys environment. From an Air Force supply management perspective, the development of the ES-S component significantly mitigated the failure of the ECSS program and eased the impact of the continuing manpower reductions by enabling the incremental fielding of new enterprise

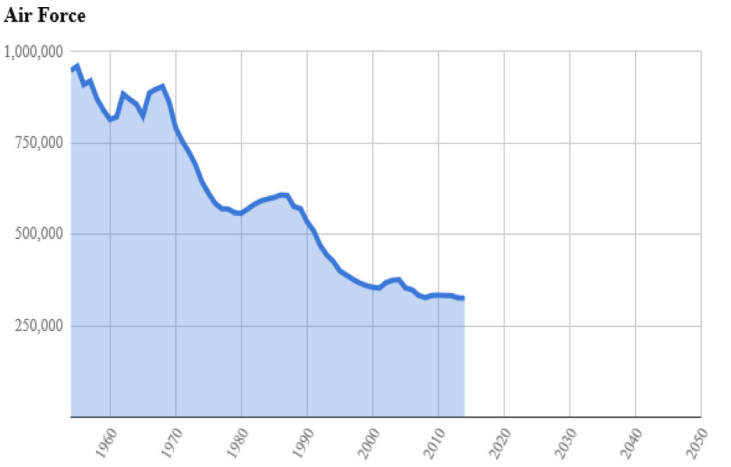

Figure 1.
Air force manpower 1954-2014

supply management functional capabilities. The ES-S development enabled three key capabilities that significantly extended Air Force supply management capabilities.

Enterprise asset and order status visibility

Prior to ES-S, SBSS users could only view the stock balances and requisition backorder status at their own bases. ES-S overcame that limitation by providing near-real-time virtual views of worldwide base-level shelf stock balances and order status and even extended user asset views to include near-real-time wholesale (Air Force Materiel Command and Defense Logistics Agency) stock balances (Reynolds *et al.*, 2007).

Platform for implementing new user capabilities

Because ES-S is hosted outside the SBSS Unisys environment, the ILS-S program office was able to incrementally implement new functional capabilities on the ES-S platform. The ES-S platform is still being used today to develop and field additional centralized enterprise management capabilities. To date, ES-S has delivered new capabilities that enable enterprise asset shipment tracking and management, data management including transaction audit records, management of warehouse validation and inventory counts, excess equipment redistribution and automatic asset sourcing and redistribution capabilities (Tosh *et al.*, 2009).

Modernized user interface screens

One of the key capabilities delivered via the ES-S component was the development of Web-accessible SBSS user interface screens that enabled worldwide users to process legacy SBSS asset management transactions via the ES-S component. In addition to providing ease of access, this ES-S feature allowed the Air Force to meet global user needs for information and transaction processing without having to incur increased licensing costs associated with the number of users allowed by the USAF Unisys contract. Additionally, the development of the modernized user interface screens eliminated user needs for a direct connection to the backend SBSS software. Separation between the system users and the legacy SBSS component decoupled supply system users from the old Unisys SBSS "green screens" that were previously used for processing SBSS transactions. Decoupling was a vital first step in making the legacy SBSS a separable ILS-S component.

Modernization of the integrated logistics system-supply application

While the ES-S component development enabled important enterprise supply management capabilities, there were fundamental issues that remained. With the cancellation of ECSS, the looming FIAR compliance requirements and the ongoing concerns about the ever-increasing operational costs of the Unisys platform, the Air Force had to move quickly to modernize the SBSS. In November 2012, the ILS-S program office conducted an analysis of alternatives for four system modernization approaches (Ellis and Harbison, 2012):

(1) *Alternative 1*: This includes re-platforming the SBSS and moving off COBOL and Unisys.

(2) *Alternative 2*: This includes re-platforming the SBSS and staying on Unisys using specialty engines.

(3) *Alternative 3*: This includes building componentized functional modules.

(4) *Alternative 4*: This includes staying on Unisys and status quo.

The Air Staff concurred with the ILS-S program office recommendation to pursue Alternative 1. The objective of the SBSS re-platform effort was to move the legacy software to a more robust, modern software language and then re-platform the new software in an open-source environment. The terminology used to describe the re-platform effort was "lift and shift". The plan required modernization of the SBSS system software, conversion of the underlying SBSS hierarchical database to a relational database, followed by a re-platforming of the updated software to an open-source hardware environment. When finished, the transformation would reflect the changes summarized in Table I.

The ILS-S program office expected that the modernization initiative would enable the migration of the system to a less costly operating environment and eventually to cloud computing. Ultimately, the program office expected to reduce annual SBSS operation costs by $15-20m per year. The re-platform effort was executed in steps. The first step was already in place via the earlier decoupling of user interaction with the legacy SBSS that was accomplished by the implementation of the ES-S modernized user interface screens. The next step was the development and implementation of an interface "wrapper" that would similarly decouple the legacy SBSS from external data system feeds. After the wrapper development, concurrent efforts were undertaken to "roll" the SBSS COBOL software to the Java software language and convert the hierarchical database to a relational database. Once all those "preparatory" efforts were completed, the modernized SBSS component could be moved to an open operating environment.

Interface wrapper

The SBSS re-platform effort involved the development of a Java software "wrapper" that decoupled the SBSS system interface processes for capturing and transferring internal, incoming and outgoing SBSS data system feeds. The left side of Figure 2 depicts how the SBSS interfaced individually with external data systems before the wrapper was implemented, while the right side illustrates the newly developed wrapper configuration. Implementation of the wrapper within the ES-S component effectively decoupled the SBSS

Table I.
Summary of SBSS modernization objectives

	Legacy SBSS	Modernized SBSS
Application software	Unisys COBOL	Java
Database software	Unisys hierarchical database	Oracle relational database
Hardware environment	Unisys mainframe	RHEL mid-tier

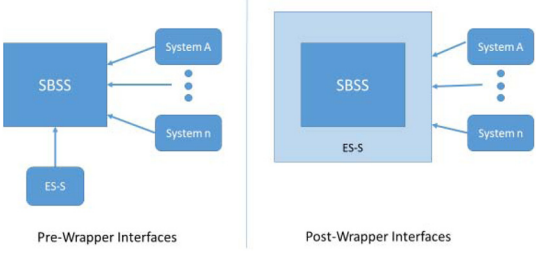

Figure 2.
SBSS wrapper implementation

from direct communication with interface partner systems. When completed, the wrapper provided three main benefits:

(1) allowed the SBSS to be migrated on a separate timetable from external interfacing data systems;

(2) enabled SBSS legacy data system interfaces to be seamlessly transitioned to modern interfaces as appropriate; and

(3) enabled a "black-box" migration approach of SBSS, focusing exclusively on data inputs and outputs without consideration of the code inside SBSS, thus eliminating "technical noise" and reducing the time required to complete the re-platform effort.

Standard Base Supply System code roll and database conversion

With the ILS-S data interface wrapper in place, the next step in the re-platform effort was modernizing the underlying software from COBOL to a more robust and modern software language. Conversions of legacy COBOL software to higher-generation software languages involve significant risk and benefit considerations. Those considerations are summarized in Figure 3 (Logapps, LLC, 2015). A key risk mitigation feature of the SBSS modernization plan was absolute avoidance of any business process reengineering (BPR) of legacy system functional processes – truly a pure "lift and shift" effort. A subtle but important benefit of restricting any BPR during the software modernization effort was that the program office was able to largely reuse existing SBSS software test cases to validate the code roll accuracy.

The code roll plan also carefully coordinated the use of military, civilian and contractor personnel in documenting the software conversion process requirement, executing the SBSS software and database conversions and rigorously testing the converted software. Key ingredients in the integrated team's success were:

- the use of automated database and code conversion tools; and
- the program office's innovative development and use of automated software testing tools.

Automated database/code conversion and automated testing tools enabled the team to rapidly "roll" and thoroughly test the modernized software by programmatically validating that the outcomes from the new software precisely matched the outcomes of the legacy SBSS component. Manual software tests often require testers to spend hours assessing the accuracy of a single data system process. The ILS-S program office team's development and application of automated testing tools enabled the team to effectively test thousands of "rolled" process outcomes in a single day.

Success

The team's successful modernization of user interface screens, wrapper development, code roll and database conversion enabled the migration of the modernized software to an open-source hardware environment. The ILS-S program office's original plan was to incrementally implement the re-platformed supply system at all worldwide bases by February 2018. The re-platformed system was fielded at 7 of the 278 Air Force bases in April and May 2017, virtually without incident. Another 70 bases were successfully migrated in June and early July. After observing the system performance at those 77 initial bases, the ILS-S program office decided to accelerate the worldwide implementation plan. By early August, the implementation had been successfully deployed to 130 bases, and

Staying With COBOL			Migrating From COBOL		
Cost Drivers	Risks	Benefits	Cost Drivers	Risks	Benefits
Increased labor costs	Potential obsolescence	Ensured successful performance	Conversion costs-code analysts, integrators, testers, server integration	System failure or information breach	Ample supply of developers
Increased mainframe costs	Lack of developers	Low risk		Loss of COBOL-specific capabilities	Freedom to develop system in many ways

Figure 3.
COBOL conversion considerations

implementation was completed across the entire Air Force in mid-September 2017 – five months ahead of schedule. The roll out of the modernized ILS-S/SBSS application was virtually seamless to worldwide users and interface partners.

The new system software enabled the Air Force to move the ILS-S/SBSS component from a proprietary Unisys environment to a Red Hat Enterprise Linux (RHEL)/Oracle environment, which greatly improved the system's sustainability. In terms of operational costs, the ILS-S program office estimated that had the software modernization effort not been undertaken and successfully completed, the fiscal year 2018 cost of operating the ILS-S/SBSS component in the proprietary Unisys environment would have been $25.5m, versus a significantly reduced annual cost for the modernized system of only $4.2m – an annual cost reduction of $21.3m.

Enabling future Air Force (AF) information technology initiatives

While the successful modernization of the ILS-S application achieved the Air Force's initial objectives in reducing operational costs and easing the effort associated with future software sustainment requirements, in some ways, the modernization effort was just a beginning. The SBSS COBOL software was organically maintained by a host of civil servants, contractors and military members over the past 35 years. Given the varying styles of those software programmers over those many years, it is not surprising that the SBSS COBOL code devolved into what is commonly referred to as "spaghetti code". As discussed earlier in this paper, the ILS-S program office significantly mitigated risk by rolling the COBOL code without any BPR. Therefore, it was inevitable that the new SBSS Java code would contain all the warts that existed within the legacy SBSS COBOL code. Additionally, even though the SBSS and ES-S components are both now implemented in the Java software language, the two components are still separated in most ways. For instance, there are a number of functional overlaps between legacy SBSS capabilities and the new ES-S enterprise asset management capability enhancements that are managed via ES-S business rules.

Now that the ILS-S modernization is completed, the program office is considering a "refactor" effort to clean-up the SBSS Java "spaghetti code" and improve the integration of the SBSS Java component with the ES-S component. A refactor of the Java code could further improve ILS-S software sustainability. Refactoring the ILS-S application could enable integration of the disparate SBSS and ES-S system modules and enable the removal of most of the legacy idioms (often referred to as "JOBOL code") from the modernized codebase. Perhaps, most importantly, the improvements resulting from a refactor effort could provide a more and more accessible and flexible foundation for addressing two important emerging Air Force IT support requirements.

Cloud migration

In January 2017, the Air Force Chief of Information Dominance and CIO, Bender (2017, p. 1), described "[. . .] the Air Force's strategic approach for the acquisition and delivery of cloud computing and storage services [. . .]" to improve mission assurance, reduce maintenance and sustainment costs and increase technical and security effectiveness. The conversion of the SBSS software to Java and the movement of the system to an open environment was a major step in preparing the ILS-S application for an eventual migration to cloud computing. In fact, as part of the modernization effort, the ILS-S program has completely migrated an instance of the modernized ILS-S application, including ES-S and the modernized SBSS component, into a commercial cloud environment. That cloud-based application is already

being used for software enhancement development and developmental testing, and could be quickly adapted as the Government test and production environment.

Logistics and supply chain information technology systems consolidation

In February 2017, Defense Secretary James Mattis (2017) began the implementation of a DoD agency overhaul which, among other initiatives, included a mandate for exploring efficiencies across core business functions, including logistics and supply chain management and cyber and IT management. This DoD mandate constituted the basis for the ILS-S program office's five-year work plan for the continued development of new functional capability enhancement efforts (Briggs, 2017). In addition to better enabling the development of new functional capabilities, a modernized and refactored ILS-S application will position the system as a potentially important foundation upon which redundant capabilities within other Air Force logistics IT systems could be consolidated to eliminate cross-system functional redundancies and reduce Air Force IT costs. As discussed earlier in this paper, the ES-S component of the ILS-S application has already been successfully used to develop, test and implement numerous new enterprise and cross-functional enterprise asset management capabilities. The ILS-S modernization path, and demonstrated successful implementation of new functional logistics capabilities, clearly shows the system's potential for further supporting the DoD IT system consolidation initiative.

Conclusion

The SBSS component of the ILS-S application was successfully transformed from a COBOL-based software application on a proprietary platform to a modernized Java-based software application system running on an open environment platform. A key preliminary step on the path to success was the isolation of the SBSS COBOL software from all direct user interfaces and all external data system communications. Once that isolation was accomplished, the COBOL code was converted to Java and implemented in place of the legacy COBOL SBSS software. The successful modernization of the Air Force SBSS application immediately reduced the annual cost of operating the ILS-S/SBSS component by $21.3M – a savings that the Air Force will realize in all future years. Additionally, the conversion of the SBSS software to Java and the movement of the system to an open environment prepared the ILS-S application for an eventual migration to cloud computing and will provide a solid IT platform for the future consolidation of Air Force logistics and supply chain management IT systems.

References

Bender, L.G.W.J. (2017), "United states air force cloud migration", SAF/CIO Memorandum for Distribution C MAJCOM-FOA-DRU-MAJCOM/CV-PEO, 23 January 2017.

Bliss, G.R. (2013), "Root cause analysis of the expeditionary combat support system program", Memorandum to the Under Secretary of Defense (AT&L), August 28, 2013.

Briggs, A.Y. (2017), "Integrated logistics systems – supply problem statement, version 1.0", ILS-S archive, Maxwell AFB, Gunter Annex, AL, 1 April 2017.

Casson, T. and Ryan, P.S. (2006), "Open standards, open source adoption in the public sector, and their relationship to Microsoft's market dominance", available at: www.logapps.com/white-papers/2015/8/5/moving-from-your-legacy-system-cobol-conversion-cost-and-consequences (accessed 15 August 2017).

CJ Pony Parts (2017), "Mustang prices through the years", available at: www.cjponyparts.com/resources/mustang-prices-through-the-years (accessed 3 August 2017).

Codlin, K.E., McGlothlin W.H., Schainblatt, A.H. and Van Horn, R.L. (1968), "Implementation of the USAF standard base supply system: a quantitative study", RAND Corporation Memorandum RM-5391-PR, July 1968.

Coleman, D. (2017), "US military personnel 1954-2014", available at: http://historyinpieces.com/research/us-military-personnel-1954-2014 (accessed 20 July 2017).

Condliffe, J. (2012), "How the US air force wasted $1 billion on a failed software plan", New York Times (accessed 10 December 2012).

Connolly, K. (2017), "Operation rolling thunder", available at: www.history.com/topics/vietnam-war/operation-rolling-thunder (accessed 3 August 2017).

Dittmer, J. (2013), "The scary truth about modernization", available at: https://fcw.com/articles/2013/12/09/comment-legacy-it-skeletons.aspx (accessed 28 July 2017).

Ellis, T.L. and Harbison, J.A. (2012), "SBSS modification analysis of alternatives", unpublished briefing to HQ USAF/A4RM, 16 November 2012.

Logapps, LLC (2015), "Moving from your legacy system – COBOL conversion, cost and consequences", available at: www.logapps.com/white-papers/2015/8/5/moving-from-your-legacy-system-cobol-conversion-cost-and-consequences (accessed 19 July 2017).

Mattis, J.N. (2017), "Establishment of cross-functional teams to address improved mission effectiveness and efficiencies in the DoD", Memorandum for Deputy Secretary of Defense, February 17 2017.

Ohnemus, R.K. (2004), "SBSS modernization efforts", Unpublished Working Paper, ILS-S archive, Maxwell AFB, Gunter Annex, AL.

Olsen, F. (1997) "Air force goes for the GOLD", available at: https://gcn.com/Articles/1997/04/14/Air-Force-goes-for-the-GOLD.aspx (accessed 21 July 2017).

Reynolds, S.B., Ohnemus, R.K., Waugh, P.B., Briggs, A.Y. and Hunter, H.H. (2007), "Interim global logistics support center supply chain management capabilities", *Air Force Journal of Logistics*, Vols 30/31 Nos 4/1, p. 118.

Spruce, F. (2002), "History of the supply X2s", Unpublished Working Paper, Feb 2002, AFLCMC/HIAR Archive, Maxwell AFB, Gunter Annex, AL.

Stross, R. (2012), "Billion-Dollar flop: air force stumbles on software plan", New York Times, (accessed 8 December 2012).

The People History (2017) "What happened in 1965 important news and events, key technology and popular culture", available at: www.thepeoplehistory.com/1965.html#tech (accessed 3 August 2017).

Tosh, G.R., Briggs, A.Y., Ohnemus, R.K., Wilson, A.I., Pratt, A.G., Talamonti, P.D., Alford, R.G., Mitchel, O.L., Hunter, S.A. and Reynolds, S.B. (2009), "Improving air force enterprise logistics management tools; enterprise solution – supply (ES-S) version 3.1 and 3.2 capability enhancements", Air Force Journal of Logistics, 22 June 2009.

Vaas, L. (2001), "Re-engineered legacy code keeps air force flying high", available at: https://gcn.com/Articles/1997/04/14/Air-Force-goes-for-the-GOLD.aspx, (accessed 20 July 2017).

Wolfe, R.E. (2017), "Computer programming", available at: http://homepage.cs.uri.edu/faculty/wolfe/book/Readings/Reading13.htm (accessed 2 August 2017).

About the authors

Mr Tommie L. Ellis is an Acquisition Program Manager assigned to the AFPEO Business Enterprise Systems (BES) Integrated Logistics Systems-Supply (ILS-S) Program Office at Maxwell Air Force Base, Gunter Annex, Alabama. He is the ILS-S Section Chief and ILS-S Program Manager.

Mr Robert A. Nicholson is an Acquisition Program Manager assigned to the AFPEO Business Enterprise Systems (BES) Integrated Logistics Systems-Supply (ILS-S) Program Office at Maxwell Air Force Base, Gunter Annex, Alabama. He is the Program Manager for the ACAT III ILS-S Standard Base Supply System (SBSS) Re-Platform Program.

Antoinette Y. Briggs is a government Senior Supply Functional Analyst. She is currently serving as the Senior Logistics Supervisor within the Air Force ILS-S program office at Maxwell Air Force Base, Gunter Annex, Alabama. She leads a team of 50 military and government contractor supply functional experts who support the ILS-S program office in developing capability that supports the global Air Force mission.

Scott A. Hunter is an Air Force employee and is the Senior ILS-S Systems Analyst. He currently serves as the Requirements Team Lead in the ILS-S program office at Maxwell Air Force Base, Gunter Annex, Alabama. He leads a team of supply functional and systems analysts who define, document and test new ILS-S functional and system enhancements. His team also provides analysis support to the development team in troubleshooting user identified system discrepancies.

James E. Harbison is the ILS-S Program Lead Engineer. His duties include strategic planning and tactical oversite for the ILS-S data system. Strategic planning covers formulating the hardware and software out-year goals for all ILS-S components (of which the SBSS re-platform is a part). Tactical oversite covers providing software engineering process guidance and validation for all ongoing ILS-S software releases. He has worked at Gunter Annex for over 27 years, with the last 10 years in the ILS-S program office specifically.

Paul S. Saladna, MBA, is an Enterprise Architect with NTT DATA Federal Services Inc. He is currently serving as the Lead Enterprise Software Architect for the Air Force ILS-S program office at Maxwell Air Force Base, Gunter Annex, Alabama. He leads a team of high-performing software developers who support the ILS-S program office in all phases of the software development life cycle. He was the Lead Architect and Technical Lead for the SBSS Code Roll project.

Michael W. Garris is a Logistics Analyst with LMI. He is currently serving as the Lead System Software Analyst with the Air Force ILS-S program office at Maxwell Air Force Base, Gunter Annex, Alabama. He supports the ILS-S program office staff by installing and maintaining system software updates and system configurations, as well as assisting Air Force users with system related problems.

Robert K. Ohnemus is a Logistics Analyst with LMI. He is a retired Chief Master Sergeant (E-9) currently serving as a Logistics Process Expert with the Air Force ILS-S program office at Maxwell Air Force Base, Gunter Annex, Alabama. He supports the ILS-S program office staff in developing, refining, documenting and fielding new and enhanced ILS-S functional capability requirements for improving worldwide Air Force supply support.

John E. O'Connor, Jr. is a retired AF colonel who spent half his active duty career in command control and logistics acquisitions, including ACAT I programs. He also served as the Director of Space and Missile Testing at AFOTEC. Since retiring, he has worked over 17 years in the private sector managing IT programs in support of the Air Force and Army. He is currently a Contractor Senior Program Manager for Windmill International, supporting the ILS-S program office.

Dr Steven B. Reynolds is a Senior Consultant with LMI. He is currently serving as a Logistics Process Expert with the Air Force ILS-S program office at Maxwell Air Force Base, Gunter Annex, Alabama. He leads a team of supply functional and IT experts who support the ILS-S program office staff in developing, refining, documenting and fielding new and enhanced ILS-S functional capability requirements for improving worldwide Air Force supply support. Steven B. Reynolds is the corresponding author and can be contacted at: sreynolds@lmi.org

For instructions on how to order reprints of this article, please visit our website:
www.emeraldgrouppublishing.com/licensing/reprints.htm
Or contact us for further details: **permissions@emeraldinsight.com**